# Module 1 – Overview of IT Industry

**Q1. Write a simple "Hello World" program in two different programming languages of your choice. Compare the structure and syntax.**

**>>>** In C language:-

```
> #include <stdio.h>
 int main()
 {
 printf("Hello, World!\n");
return 0;
 }
```

In Python:-

```
> print("Hello, World!")
```

**Comparison of Structure and Syntax:**

**—>** C requires a main function as entry point. Code is organized within blocks using curly braces { }.

**—>** Python requires No explicit main function needed. Code execution starts from the top.

**—>** In C Semicolons ; are used to terminate the statements.

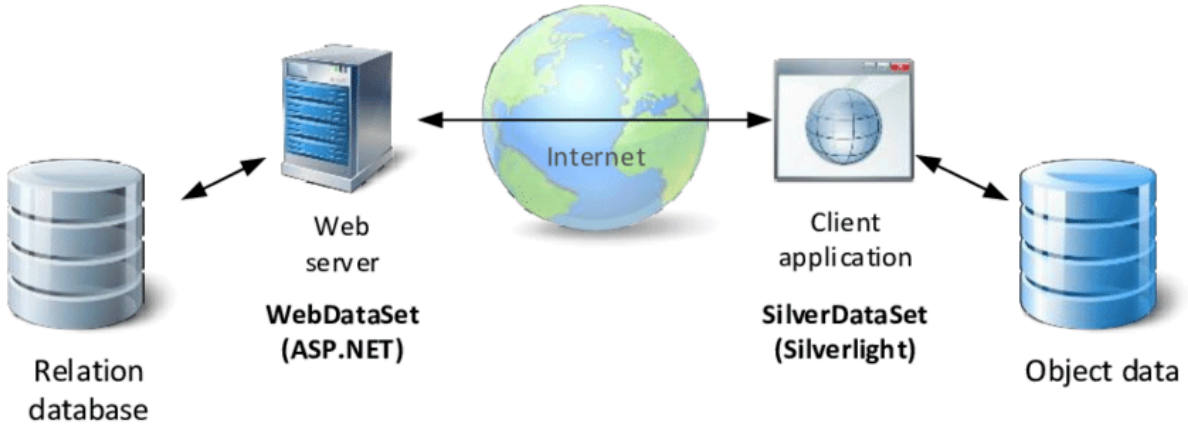
—> In python no semicolons require at the end of statements.

—> C uses  functions like printf( ) and scanf( ).

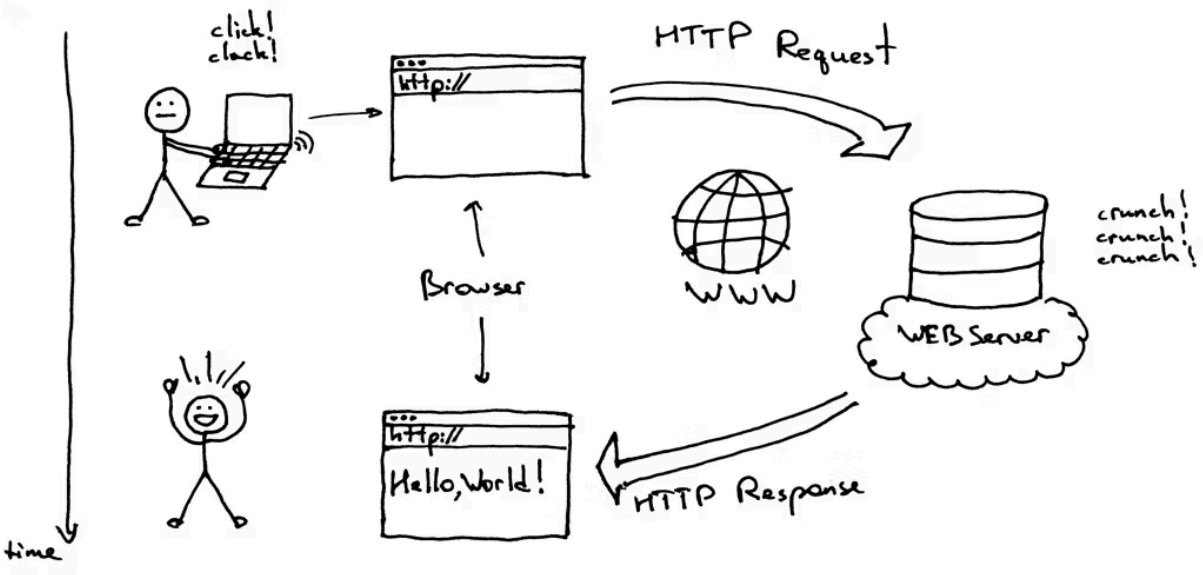
—> Python uses print and input functions.

## Q2.Research and create a diagram of how data is transmitted from a client to a server over the internet.

>>>




## Q3. Design a simple HTTP client-server communication in any language.




**Q4. Research different types of internet connections (e.g., broadband, fiber, satellite)and list their pros and cons.**

**1. Broadband (DSL & Cable)**

➢Description: Uses telephone lines (DSL) or coaxial cables (Cable) to provide internet access.

**Advantages:-**

➔Widely available

➔ More affordable than fiber

➔ Reliable for general use

**Disadvatages:-**

➔Slower speeds compared to fiber
➔Affected by network congestion (especially cable)
➔ Distance from the provider affects DSL speed

**2. Fiber Optic**

➢Description: Uses fiber-optic cables to deliver high-speed internet.

**Advantages:-**

➔ Extremely fast speeds (up to 10 Gbps)
➔ Low latency (ideal for gaming, streaming, and video calls)
➔ More reliable than cable or DSL

**Disadvantages:-**

➔Expensive installation costs
➔Limited availability in rural areas
➔Requires special infrastructure

**3. Satellite**

➢Description: Internet is beamed from satellites to a dish installed at the user's location.

**Advantages:-**

➔Available in remote and rural areas

➔No need for underground cables
➔Can provide a stable connection where other options are unavailable

**Disadvantages:-**

➔High latency (lag)
➔ Expensive compared to wired internet
➔ Weather can affect the connection

**4. Fixed Wireless**

➢Description: Uses radio signals from towers to provide internet access.

**Advantages:-**

➔Good for rural areas without fiber or cable
➔No need for underground wiring
➔Faster than satellite in some cases

**Disadvantages:-**

➢Line-of-sight issues (obstructions like trees/buildings can affect signals)
➢Slower speeds than fiber and cable
➢Weather conditions can disrupt service

**5. Mobile (4G/5G)**

➢Description: Provides internet through cellular networks (4G LTE & 5G).

**Advantages:-**

➢Portable and doesn’t require installation
➢5G offers extremely high speeds
➢Available almost everywhere with mobile coverage

**Disadvantages:-**

➔Data caps and throttling from carriers
➔Speed depends on network congestion
➔More expensive than wired connections

**6. Dial-Up (no longer useful  but still in use in some areas)**

➢Description: Uses a telephone line to connect to the internet.

**Advantages:-**

➔Extremely cheap
➔Can work where broadband isn’t available

**Disadvantages:-**

➔Very slow speeds (max 56 Kbps)
➔Cannot use phone and internet simultaneously
➔Outdated technology

## Q5. Identify and explain three common application security vulnerabilities. Suggest possible solutions.

### 1. SQL Injection (SQLi):-

SQL injection occurs when an attacker manipulates an application's database query by injecting malicious SQL code through input fields. This can lead to unauthorized access, data leakage, or database modification.

### Example:

SELECT * FROM users WHERE username = 'admin' OR '1'='1' --' AND password = 'password';

The OR '1'='1' condition always evaluates to true, potentially bypassing authentication.

Solution:

Use prepared statements (parameterized queries).
Sanitize and validate user inputs.
Implement least privilege principle for database access.

Use Web Application Firewalls (WAFs) to detect and block SQLi attempts.

---

## 2. Cross-Site Scripting (XSS)

Description:
XSS allows attackers to inject malicious JavaScript into a web application, which is then executed in users' browsers. This can be used to steal cookies, redirect users, or deface websites.

Example:

<script>alert('Hacked!');</script>

If an input field does not properly sanitize user input, an attacker could inject this script, causing it to execute on other users' browsers.

Solution:

>Escape user inputs before rendering them in HTML
>Use Content Security Policy (CSP) headers to restrict JavaScript execution
>Implement input validation and sanitization
>Use security libraries like DOMPurify to clean user input

### 3. Broken Authentication & Session Management

Description:
This vulnerability occurs when authentication mechanisms (such as login systems and session handling) are weak, allowing attackers to hijack sessions, brute-force passwords, or gain unauthorized access.

Example:

- ➢Storing passwords in plaintext
- ➢Using weak session cookies without expiration

Solution:

Use strong password hashing algorithms (e.g., bcrypt, Argon2)
Implement Multi-Factor Authentication (MFA)
Set secure and HttpOnly flags on session cookies
Implement account lockout mechanisms to prevent brute-force attacks

## Q6. Identify and classify 5 applications you use daily as either system software orapplication software.

**1. Operating System (Windows/macOS/Linux) – System Software**

- Category: System Software
- Purpose: Manages hardware and software resources, providing a platform for other applications to run.

**2. Web Browser (Google Chrome, Mozilla Firefox) – Application Software**

- **Category: Application Software**
- **Purpose: Allows users to browse the internet, access websites, and use web applications.**

**3. Microsoft Word (or Google Docs) – Application Software**

- Category: Application Software
- Purpose: Used for creating, editing, and formatting documents.

4. Antivirus Software (Windows Defender, Avast) – System Software

- Category: System Software
- Purpose: Protects the system from viruses, malware, and security threats.

**5. Media Player (VLC, Windows Media Player) – Application Software**

- Category: Application Software
- Purpose: Plays audio and video files for entertainment or work purposes.

## Q7. Create a case study on the functionality of the presentation, business logic, and data access layers of a given software system.

Case Study: Functionality of the Presentation, Business Logic, and Data Access Layers in an E-Commerce Application

**Introduction:**

In this case study, we will explore the functionality of three crucial layers in a typical software system — Presentation Layer, Business Logic Layer, and Data Access Layer — through the example of an e-commerce application. These layers work together to create a seamless and efficient user experience, support business processes, and manage data flow between the application and the underlying database.

**1. Presentation Layer:**

**Overview:** The presentation layer is the topmost layer responsible for handling user interaction and presenting information to the end-user. It acts as an interface

between the user and the system, allowing the user to input data and view output.

**Functionality:**

- **User Interface (UI):** This layer provides the graphical elements that users interact with, such as forms, buttons, navigation menus, and product listings. For instance, in an e-commerce application, the UI displays categories of products, product details, shopping cart items, and the checkout process.
- **Input Validation:** Before any data is passed to the business logic layer, the presentation layer ensures that the data entered by users is valid. For example, in the checkout process, it ensures that the user's shipping address is properly formatted.
- **Displaying Results:** The presentation layer formats and displays the output received from the business logic layer, such as order confirmation, product search results, or error messages.
- **Technologies Used:** HTML, CSS, JavaScript, and frameworks like React, Angular, or Vue.js are used for building the UI. Server-side technologies like ASP.NET or Node.js can also render dynamic content.

**Example:**

- When a user adds an item to their cart, the presentation layer dynamically updates the cart display to reflect the new item, showing details like the product name, price, and quantity.

**2. Business Logic Layer:**

**Overview:** The business logic layer, also known as the application layer, processes the core business rules of the application. This layer is responsible for decision-making, calculations, and orchestrating the flow of data between the presentation layer and data access layer.

**Functionality:**

- **Business Rules:** This layer defines the rules for how data is processed. For example, in an e-commerce system, the business logic layer determines how discounts should be applied based on user loyalty, seasonal promotions, or specific product categories.
- **Data Validation:** Although some validation happens in the presentation layer, more complex validation (such as checking stock levels or ensuring that a product is available in the chosen shipping region) occurs in the business logic layer.
- **Transaction Management:** The business logic layer is responsible for managing transactions between the user and the data access layer. For instance,

when the user places an order, the system processes the payment, deducts stock from inventory, and records the order details.
- **Technologies Used:** Typically implemented using backend technologies such as Java (Spring), C# (ASP.NET), Python (Django), or Node.js.

**Example:**

- When the user proceeds to checkout, the business logic layer calculates the total price (including taxes, discounts, and shipping) and applies any promotional codes the user entered. It then passes this data to the data access layer to record the order.

**3. Data Access Layer:**

**Overview:** The data access layer (DAL) is responsible for interacting with the database or other data storage systems. It abstracts the underlying data structure, providing a simplified interface to retrieve, modify, and persist data without exposing complex queries to the business logic or presentation layers.

**Functionality:**

- **Data Retrieval:** The DAL handles all the data queries required by the business logic layer. For instance, when a user searches for a product, the DAL

retrieves the relevant product details from the database.

- **Data Insertion and Updates:** When a user places an order, the DAL inserts order details into the database and updates stock levels. It may also update customer records, shipping information, and payment status.
- **Data Abstraction:** The DAL hides the complexities of database operations from the rest of the application, allowing developers to work with high-level objects and APIs. This layer also optimizes database access for efficiency.
- **Transaction Management:** The DAL ensures that database transactions are executed properly, maintaining data integrity, consistency, and isolation (ACID properties).
- **Technologies Used:** SQL databases (MySQL, PostgreSQL, Oracle), NoSQL databases (MongoDB), ORMs (Object-Relational Mappers) like Hibernate or Entity Framework.

**Example:**

- When the business logic layer calculates the total order amount, it requests the DAL to fetch the product prices, inventory status, and shipping options. Upon order completion, the DAL updates the database with the order and payment status.

**Interactions Between the Layers:**

- **Presentation to Business Logic Layer:** The presentation layer sends requests to the business logic layer (e.g., to calculate the total cost of an order or validate a coupon). This is done via API calls, which are processed by the business logic.
- **Business Logic to Data Access Layer:** The business logic layer queries the DAL for data, such as product information or customer account details. The DAL then communicates with the database to retrieve, insert, or update the necessary data.
- **Flow of Data:** When the user interacts with the system, data flows from the presentation layer to the business logic layer, which processes it and requests data from the data access layer. The DAL responds with the required data, and the business logic layer returns the processed data back to the presentation layer for display to the user.

## Q8.Simulate HTTP and FTP requests using command line tools (e.g., curl).

>> For HTTP requests using curl:

Basic GET request.

Code- curl https://www.example.com

Post request Code- curl -X POST -H “Content-Type: application/json” -d ‘{“key”: “value”}’

https://api.example.com/endpoint

Specifying headers Code- curl -H “Authorization: Bearer your_token” https://api.example.com/protected-resource

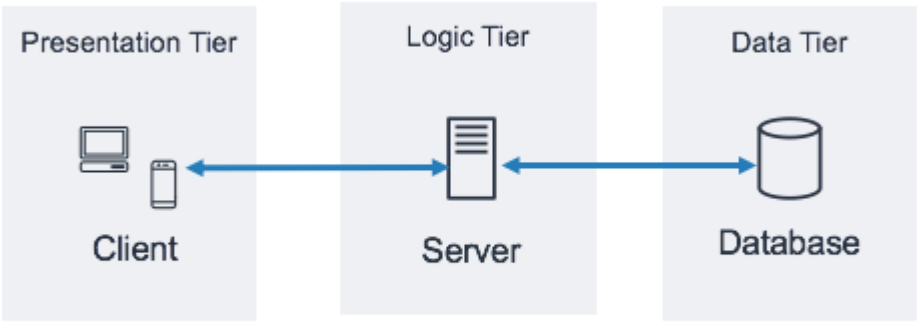
For FTP requests using curl:

List directory contents Code- curl -u username:password -T – ftp://server.com/path/to/directory

Upload a file Code-curl -T your_file.txt -u username:password ftp://server.com/path/to/upload

Download a file Code-curl -O -u username:password ftp://server.com/path/to/file.txt

**Q9.Design a basic three-tiersoftware architecture diagram for a web application.**




## Q10.Create a case study on the functionality of the presentation, business logic, and dataaccess layers of a given software system.

➢In modern software architecture, applications are often designed using a three-tier architecture, comprising the Presentation Layer, Business Logic Layer, and Data Access Layer. This case study examines how these layers function in an online e-commerce platform.

### System Overview

The case study focuses on an e-commerce web application designed to facilitate online shopping. The system includes features such as user authentication, product browsing, order processing, and payment integration.

### 1. Presentation Layer (PL)

**Functionality:** The Presentation Layer is the front-end interface that users interact with. It is responsible for rendering the user interface (UI), collecting user inputs, and displaying data received from the Business Logic Layer.

**Technologies Used:**

- HTML, CSS, JavaScript (React.js)
- RESTful API calls
- Bootstrap for styling

**Example:** A user logs into the platform and browses available products. The UI fetches product details from the Business Logic Layer and displays them dynamically.

### 2. Business Logic Layer (BLL)

**Functionality:** The Business Logic Layer processes user inputs, enforces business rules, and interacts with the Data Access Layer to retrieve and update data. This layer ensures that only valid transactions occur.

**Technologies Used:**

- Node.js with Express.js
- Java-based Spring Boot
- Python with Django

**Example:** When a user adds a product to their cart, the BLL validates product availability, calculates pricing (including taxes and discounts), and updates the session data accordingly.

### 3. Data Access Layer (DAL)

**Functionality:** The Data Access Layer handles direct interactions with the database. It ensures data integrity, performs CRUD (Create, Read, Update, Delete) operations, and optimizes queries.

**Technologies Used:**

- MySQL / PostgreSQL
- MongoDB (NoSQL option)
- Object-Relational Mapping (ORM) tools like Sequelize, Hibernate, or Entity Framework

**Example:** When the Business Logic Layer requests product details, the DAL executes a SQL query to fetch the relevant product information from the database and returns it securely.

## Q11.Explore different types of software environments (development, testing, production).Set up a basic environment in a virtual machine.

Software environments play a crucial role in the software development lifecycle. The three main types are:

- **Development Environment**
  - Used by developers to write and test code.
  - May include IDEs (e.g., VS Code, IntelliJ), local databases, and debugging tools.
  - Frequent changes and updates occur in this environment.
- **Testing (Staging) Environment**
  - Used for quality assurance (QA) and bug detection before deployment.
  - Often mirrors the production environment but with test data.
  - Includes unit, integration, and user acceptance testing (UAT).
- **Production Environment**
  - The live environment where end-users interact with the software.
  - Stability and security are crucial.
  - Changes require careful deployment to avoid downtime.
- **Setup:**

1.Install Virtual Machine Software

2.Create a New Virtual Machine

3.Install the Operating System

4.Set Up Development Tools (For Development Environment)

5.Set Up Testing Tools (For Testing Environment)

6.Set Up Production Environment (For Production Environment)

7.Finalize and Test

## Q12.Write and upload your first source code file to Git hub.

1. Create a GitHub account

2. Create a new repository for your project.

3. Upload a files to your project's repository

## Q13.Create a Git hub repository and document how to commit and push code changes.

→ git push: Pushes changes from the local repository to GitHub. Git commit -m "message":

## Q14. Create a student account on Git hub and collaborate on a small project with A classmate.

1.Go to the repository you just created.

2.Click on the Settings tab in the repository.

3.Scroll down to the Collaborators section on the left sidebar.

4.Under Manage access, click Invite a collaborator.

5.Search for your classmate’s GitHub username and invite them to your repository.

6.Your classmate will receive an invitation via email or GitHub notifications. They need to accept the invitation to collaborate on the repository.

## Q15.Create a list of software you use regularly and classify them into the following categories: system, application, and utility software.

Here's a categorized list of software you might use regularly, based on your work as a graphic designer, 3D animator, video editor, and advertisement designer:

### 1. System Software (Manages hardware and system operations)

- Windows / macOS / Linux – Operating system
- Drivers (GPU, Printer, Wacom Tablet, etc.) – Hardware management
- Firmware Updates – For BIOS, peripherals, and devices

## 2. Application Software (User-focused software for work and creativity)

### Graphic Design & Illustration

- Adobe Photoshop – Image editing and digital painting
- Adobe Illustrator – Vector graphics design
- CorelDRAW – Vector illustration

### 3D Animation & Modeling

- Blender – 3D modeling, animation, and rendering
- Autodesk Maya – Professional 3D animation
- Cinema 4D – Motion graphics and 3D design
- ZBrush – Sculpting and 3D detailing

### Video Editing & Motion Graphics

- Adobe Premiere Pro – Video editing
- Adobe After Effects – Motion graphics and VFX
- DaVinci Resolve – Color grading and video editing

---

## 3. Utility Software (Helps optimize and maintain the system)

- CCleaner – System cleanup and optimization
- WinRAR / 7-Zip – File compression and extraction
- OBS Studio – Screen recording and streaming
- Notion / Evernote – Note-taking and organization

- Google Drive / Dropbox – Cloud storage and backup
- NVIDIA GeForce Experience – GPU driver updates and optimization
- HandBrake – Video conversion and compression

## Q16.Follow a GIT tutorial to practice cloning, branching, and merging repositories.

1.Cloning: git clone https://github.com/yourusername/git-tutorial.git

2.Branching: Git checkout -b feature-branch

3.Merging: git merge feature-branch

## Q17.Write a report on the various types of application software and how they Improve productivity.

**1.Productivity Software:**Tools like Microsoft Office and Google Workspace streamline document creation, data analysis, and collaboration.

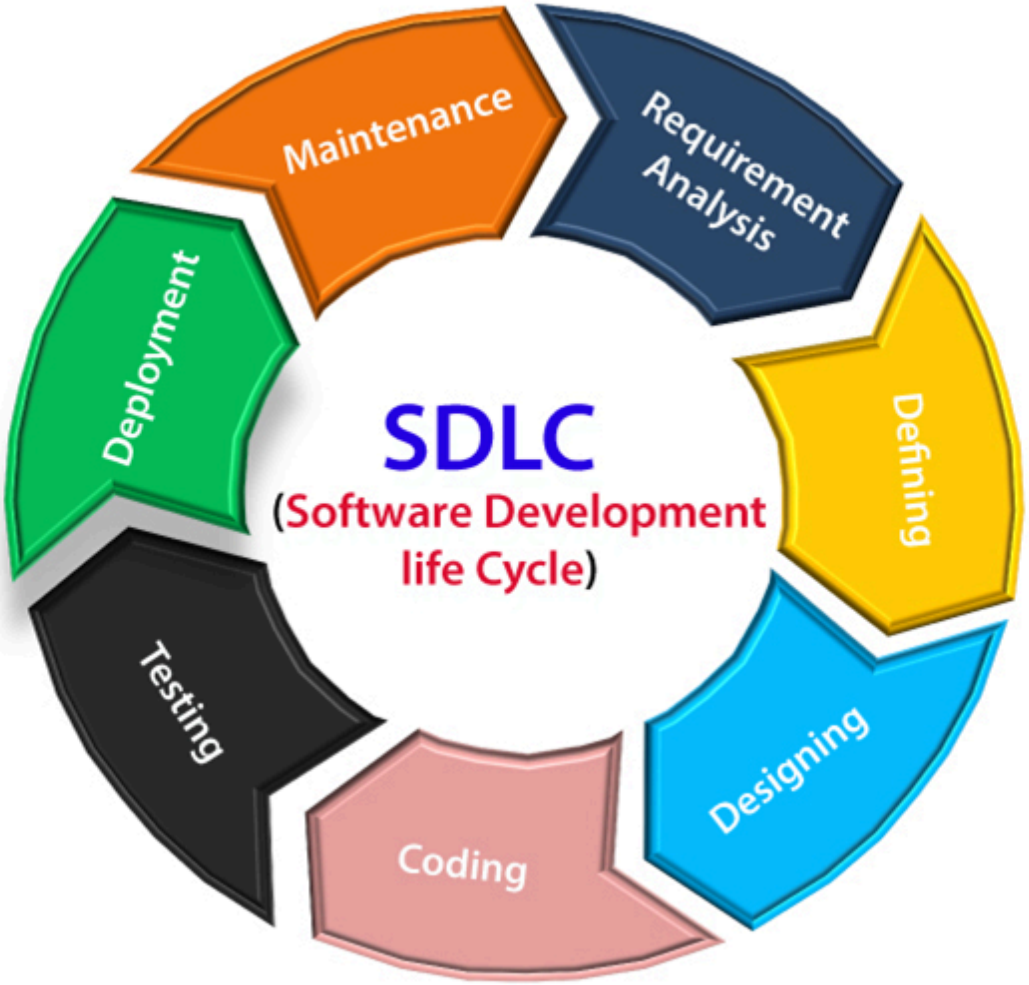
**2.Utility Software:** Programs like antivirus and file management tools optimize system performance and prevent downtime.

**3.Web Browsers:** Browsers facilitate quick access to online resources, supporting research and communication.

**4.Communication Software:** Platforms like Zoom and Slack enable seamless remote collaboration, saving time and resources.

**5.education software :** Platforms like Moodle and Coursera provide on-demand learning, reducing time spent on traditional education.

## Q18. Create a flowchart representing the Software Development Life Cycle (SDLC).



## Q19. Write a requirement specification for a simple library management system.

**1.user management:** Handel library members,staff with secure login system

**2.book management:**admin can add,update or remove book.user can search for the book and check for the avilability.

**3.borrowing& returning:** track issued book,due dates.

## Q20.Perform a functional analysis for an online shopping system.

An online shopping system enables users to browse, select, purchase, and manage products through a web-based platform. The system facilitates interactions between customers, sellers, and administrators while ensuring smooth transactions and user satisfaction. This analysis outlines the primary functions of an online shopping system.

➢ **User Management**

- User Registration and Login
- Profile Management
- Password Recovery

- Role-Based Access Control (Customers, Sellers, Admins)

➢ **Product Catalog Management**

- Product Listing and Categorization
- Product Search and Filters
- Product Details and Reviews
- Inventory Management

➢**Shopping Cart Functionality**

- Add/Remove Items from Cart
- Update Item Quantities
- Save Cart for Future Purchase
- Apply Coupons and Discounts

➢**Order Processing**

- Order Placement and Confirmation
- Order Tracking
- Payment Gateway Integration
- Order History and Invoice Generation

➢**Payment Management**

- Multiple Payment Methods (Credit/Debit Card, PayPal, Digital Wallets)
- Secure Transactions and Encryption
- Refund and Cancellation Processing

➢**Customer Support**

- Live Chat Support
- FAQs and Help Center
- Contact Forms and Ticketing System

➢**Seller Management**

- Seller Registration and Approval
- Product Upload and Management
- Sales Analytics and Reports

➢**Admin Management**

- User and Seller Moderation
- Product Approval and Categorization
- Order and Payment Oversight
- System Analytics and Performance Monitoring

➢**Security Features**

- User Authentication and Authorization
- Secure Payment Processing
- Data Encryption and Privacy Compliance

➢**Additional Features**

- Personalized Recommendations
- Wishlist and Favorites
- Notifications and Alerts
- Multi-Language and Multi-Currency Support

## 21.Design a basic system architecture for a food delivery app.

Introduction A food delivery app connects customers with restaurants and delivery personnel to facilitate the ordering, payment, and delivery of food. The system should ensure scalability, security, and a seamless user experience.

## System Components

➔**User Interfaces**

➔**Customer App (iOS/Android/Web)**

- ◆User registration and login
- ◆Restaurant and menu browsing
- ◆Order placement and tracking
- ◆Payment processing
- ◆Reviews and ratings

➔**Restaurant Partner App (iOS/Android/Web)**

- ◆Restaurant profile and menu management
- ◆Order acceptance and preparation tracking
- ◆Earnings and order history

➔**Delivery Partner App (iOS/Android/Web)**

- ◆Order assignments and navigation
- ◆Real-time status updates
- ◆Earnings and delivery history

➔**Admin Dashboard (Web-based)**

- ◆User and restaurant management
- ◆Order and payment oversight
- ◆Reports and analytics

## Backend System

➔API Gateway: Manages communication between frontend and backend.

➔Authentication & Authorization: Secure login and role-based access.

➔Order Management System (OMS): Handles order placement, tracking, and status updates.
➔Payment Processing Module: Integrates with payment gateways for transactions.
➔Restaurant Management System: Manages menu, availability, and order processing.
➔Delivery Management System: Assigns and tracks delivery personnel.
➔Notification System: Sends real-time updates via push notifications, email, and SMS.
➔Analytics & Reporting Module: Generates insights on sales, performance, and user behavior.
➔**Database Architecture**

- **User Database:** Stores customer, restaurant, and delivery partner information.
- **Order Database:** Tracks order details, status, and history.
- **Menu Database:** Stores restaurant menus, pricing, and availability.
- **Payment Database:** Manages transaction history and refunds.
- **Review & Rating Database:** Stores user feedback on restaurants and delivery personnel.

**External Integrations**

- **Payment Gateways:** Stripe, PayPal, or local payment providers.

- **Mapping & Navigation:** Google Maps, OpenStreetMap.
- **SMS & Email Services:** Twilio, SendGrid.
- **Cloud Storage:** AWS, Google Cloud Storage for media and documents.

**System Workflow**

- A customer places an order via the app.
- The restaurant receives and confirms the order.
- A delivery partner is assigned based on location and availability.
- The delivery partner picks up the food and updates the status in real-time.
- The customer receives notifications throughout the process.
- Payment is processed securely, and the order is marked complete.
- The customer can leave a review and rating.

## Q22.Develop test cases for a simple calculator program.

To test case calculator you can test.

1. Basic arithmetic:addition , subtraction , multiplication,division .

2.complax case: divide by zero Multiple calculation (ex:2+3*5)

3.handeling decimal input .error handling : invalid operator, alphabets.

## Q23.Document a real-world case where a software application required Critical maintenance.

In 2017 Microsoft face critical issue with a windows 10 creator update, which cause system crashes, performance issue and driver compatibility problems for users

Response: Microsoft provided a rollback option, released patch updates to fix bugs and security flaws, and improved its update testing process.

## Q24. Create a DFD for a hospital management system.